1変数代数方程式の一つの近接根クラスタに含まれる近接根の計算

AUTHOR(S):

照井, 章; 佐々木, 建昭

CITATION:

照井, 章 ...[et al]. 1変数代数方程式の一つの近接根クラスタに含まれる近接根の計算. 数理解析研究所講究録 2005, 1456: 27-34

ISSUE DATE:

2005-11

URL:

http://hdl.handle.net/2433/47852

数理解析研究所講究録 1456 巻 2005 年 27-34

1変数代数方程式の一つの近接根クラスタに含まれる近接根の計算

照井 章 * 佐々木 建昭 †

AKIRA TERUI TATEAKI SASAKI

筑波大学大学院 数理物質科学研究科

GRADUATE SCHOOL OF PURE AND APPLIED SCIENCES, UNIVERSITY OF TSUKUBA

Abstract

複素係数1変数多項式 $P(x)$ は近接根クラスタ（複数でもよい）を持ち、クラスタは他根から十分に分離できるとする。このクラスタに含まれる全根を同時に、他の根を計算しないで、効率良く計算する算法を提案する。算法は Durand-Kerner 法を近接根用に改変したものであり、非常に簡単である。末尾に実験例を示す。

1 はじめに

古来より、代数方程式の解法として非常に多くの方法が提案されており、近接根を含まない多項式に対しては、たとえ悪条件であっても見事に解く方法も発表されている（Fortune [2], 他）。また、与えられた多項式を、複素円盤の内部と外部にそれぞれ根を持つ二つの多項式の積に分解する高速算法により、計算量も決定されつつある（Pan [7], 他）。しかし、近接根を含む多項式に関してはまだ研究の余地がある。Yakoubsohn [11] は、近接根に対する Newton 法の収束定理を証明したが、その算法は効率的とは言えない。最近、Sakurai らが近接根用の巧妙な算法を発表した [5]。その方法は近接根を桁落ちなく計算するが、算法の複雑さが難点である。

一方、近接根を含まない多項式に関しては、Durand-Kerner 法 ([1], [4]) などの簡潔かつ効率的な方法がある。同様に、一つの近接根クラスタ内の全ての根を同時に、他の根を一切計算することなく、しかも桁落ちを起こすことなく、効率良く計算することはできないものであろうか？ 本稿ではそのような方法を提案する。この方法では、与多項式は変換され、近接根以外の根の影響は多項式の商として取り込まれる。その際、変換された多項式および反復の度に計算される商は、（クラスタの近接度に依存する）一定次数分だけ計算すればよく、結果として高次多項式に対しては著しい効率化を達成する。たとえば、10 個の近接根を含むクラスタを持つ 1000 次の多項式の場合でも、変換後の多項式は 20〜25 次分を、商は 10〜15 次分を計算すればよい。我々の方法は近接根の近くの根の影響だけを取り込んで計算を行う方法である、と言えよう。

以下、本稿では次の内容を述べる。**2** では算法の基本的なアイデアについて述べる。**3** では近接根用 Durand-Kerner 法について述べる。**4** では実際に算法を実行する上での注意点について述べ、計算例を掲げる。**5** では実験結果を示す。

*terui@math.tsukuba.ac.jp

†sasaki@math.tsukuba.ac.jp

2 算法の基本的アイデア

我々の算法は次の三つのステップからなる。

1. 近接根クラスタの位置 ρ と近接度 2γ、クラスタ中の根の個数 m を決定する。ここで、近接度とは [近接根クラスタの広がり]/[根全体を含む円盤の半径] で定義する。
2. ターゲットとするクラスタを原点に移動し、$x \mapsto cx$ なるスケール変換で、クラスタを半径1程度の大きさに拡大する（このステップは多倍長計算を必要とする）。
3. 一つのクラスタ中の全根だけを扱うように Durand-Kerner 法を改変し、反復的に近似解を計算する（このステップは固定長計算で十分である）。

この算法は、3) の反復算法が不明な点を除けば、実に平凡かつ簡単である。3) の反復算法も、我々が以前に発表した論文（実根用 Durand-Kerner 法 [10]：実多項式の実根だけを同時反復法で計算する方法）と全く同じである。こんな簡単な算法がうまく働くのか? と思われようが、後で見るように、実にうまく働くのである。

まず、ステップ1について。我々の算法では、ターゲットとする近接根クラスタの位置 ρ、近接度 2γ、およびクラスタに含まれる根の個数 m、が必要である。これらの情報は、近似無平方分解（[6], [8] を参照）で得られるので、本稿ではこれらの情報は既知として議論を進める。

近接根がクラスタをなすと言っても、そのクラスタが他の根から区別できなければ、実際的にクラスタをなすとは言えず、我々の算法は破綻する。近接根クラスタが他の根から明確に区別できるかどうかについて、著者らは数年前に簡潔な定理を得ており [9]、その定理で区別できることを確認しておく。

次に、ステップ2について。与式を $P(x)$ とする：

$$P(x) = c_n x^n + \cdots + c_m x^m + \cdots + c_0. \tag{1}$$

説明の便宜上、多項式 $P(x+\rho)$ の根は原点を中心とする半径2の複素円盤内に含まれるとする（この仮定は実際の計算では必要ないが、この仮定の下では近接根クラスタの広がりは約 2γ となる）。Terui-Sasaki の近接根分離定理によれば、$P(x+\rho)$ の m 次未満の項の係数は、次数が一つ下がれば約 γ 倍減少する。ただし、$\gamma < 1/9$ である [9]。このことを念頭に、スケール変換 $P(x) \mapsto \tilde{P}(x) = \eta P(\gamma(x+\rho))$ を考えよう（η は $\tilde{P}(x)$ のノルム規格化のための数値である）：

$$\tilde{P}(x) = \tilde{c}_n x^n + \cdots + \tilde{c}_m x^m + \cdots + \tilde{c}_0. \tag{2}$$

近接根に対応する $\tilde{P}(x)$ の m 個の根は原点を中心とする半径2程度の複素円盤内にあるので、

$$\tilde{c}_m = 1, \quad \max\{|\tilde{c}_{m-1}|, \cdots, |\tilde{c}_0|\} \approx 1, \tag{3}$$

となるように決めることができる。このとき、高次項の係数は次のようになる。

$$\tilde{c}_{m+j} = o(\gamma^{j-1}) \quad (j = 1, 2, \ldots, n-m). \tag{4}$$

この性質は実際の算法において非常に重要な役割をはたす。

3 近接根用 Durand-Kerner 法

式 (2) の $\tilde{P}(x)$ に対して、論文 [10] のアイデアに従い、Durand-Kerner 法を改変する。

$\tilde{P}(x)$ に対し、方程式 $\tilde{P}(x)=0$ の全根を Durand-Kerner の 2 次法で求める場合、根の初期値 $z_1^{(0)},\ldots,z_n^{(0)}$ を与え、次の反復式によって、根 $\zeta_1,\ldots,\zeta_n$ に対応する ν 回反復後の近似値 $z_1^{(\nu)},\ldots,z_n^{(\nu)}$ を計算する。

$$z_j^{(\nu+1)}=z_j^{(\nu)}-\frac{\tilde{P}(z_j^{(\nu)})}{\tilde{c}_n\tilde{Q}'(z_j^{(\nu)})},\quad \tilde{Q}(x)=\prod_{j=1}^{n}(x-z_j^{(\nu)}),\quad j=1,\ldots,m. \tag{5}$$

ここで、求める近接根を $\zeta_1,\ldots,\zeta_m$、その近似値をそれぞれ $z_1^{(\nu)},\ldots,z_m^{(\nu)}$ $(m<n)$ とする。本稿の算法は、これ以外の根の近似値 $z_{m+1}^{(\nu)},\ldots,z_n^{(\nu)}$ を用いないため、式 (5) の $\tilde{Q}(x)$ を構成することができない。

ところで、すべての近似値 $z_j^{(\nu)}$ が根 ζ_j に十分近づくと、$\tilde{Q}(x)$ は $\tilde{P}(x)$ に近づく（係数同士が互いにほぼ等しくなる）。そこで、本稿の算法では, $\tilde{Q}(x)$ のうち, 求める近接根以外の近似値からなる因子 $\prod_{j=m+1}^{n}(x-z_j^{(\nu)})$ を、$P(x)$ を近接根の近似値からなる因子 $\prod_{j=1}^{m}(x-z_j^{(\nu)})$ で除した商 $\mathrm{quo}(\tilde{P}(x),\prod_{j=1}^{m}(x-z_j^{(\nu)}))$ で近似する。すなわち、本稿における Durand-Kerner 法の反復式として、次式を採用する。

$$z_j^{(\nu+1)}=z_j^{(\nu)}-\frac{\tilde{P}(z_j^{(\nu)})}{Q_\nu'(z_j^{(\nu)})},\quad Q_\nu(x)=\prod_{j=1}^{m}(x-z_j^{(\nu)})\cdot\mathrm{quo}(\tilde{P}(x),\prod_{j=1}^{m}(x-z_j^{(\nu)})),\quad j=1,\ldots,n. \tag{6}$$

反復式 (6) は次の収束定理をみたす [10]。

定理 1

$\zeta_1,\ldots,\zeta_m$ を方程式 $\tilde{P}(x)=0$ の根とし、互いに異なるものとする。$z_1^{(\nu)},\ldots,z_m^{(\nu)}$ をそれぞれ $\zeta_1,\ldots,\zeta_m$ の近似値とし、$z_j^{(\nu)}$ は ζ_j に十分近いものとする。このとき、反復式 (6) は 2 次収束する。

証明 近似値 $z_j^{(\nu)}$ の誤差を $\varepsilon_j^{(\nu)}=z_j^{(\nu)}-\zeta_j$ $(j=1,\ldots,m)$ とおくと、次式を得る。

$$\varepsilon_j^{(\nu+1)}=z_j^{(\nu+1)}-\zeta_j=\varepsilon_j^{(\nu)}-\frac{\tilde{P}(z_j^{(\nu)})}{Q_\nu'(z_j^{(\nu)})}=\varepsilon_j^{(\nu)}-\varepsilon_j^{(\nu)}\frac{\prod_{k=1,\neq j}^{n}(z_j^{(\nu)}-\zeta_k)}{Q_\nu'(z_j^{(\nu)})} \tag{7}$$

一方, $R(x)=\prod_{k=m+1}^{n}(x-\zeta_k)$, $\varepsilon^{(\nu)}=\max\{|\varepsilon_1^{(\nu)}|,\ldots,|\varepsilon_m^{(\nu)}|\}$ とおくと, $Q_\nu'(z_j^{(\nu)})$ は次式で表される.

$$Q_\nu'(z_j^{(\nu)})=\prod_{k=1,\neq j}^{m}(z_j^{(\nu)}-z_k^{(\nu)})\prod_{k=m+1}^{n}(z_j^{(\nu)}-\zeta_k)+(\varepsilon_1^{(\nu)}+\cdots+\varepsilon_m^{(\nu)})R'(z_j^{(\nu)})+O((\varepsilon^{(\nu)})^2). \tag{8}$$

式 (8) を用いることにより, 式 (7) の最右項の有理式は次の通り変形できる.

$$\begin{aligned}\frac{\prod_{k=1,\neq j}^{n}(z_j^{(\nu)}-\zeta_k)}{Q_\nu'(z_j^{(\nu)})}&=\prod_{k=1,\neq j}^{m}\left(\frac{z_j^{(\nu)}-\zeta_k}{z_j^{(\nu)}-z_k^{(\nu)}}\right)+O(\varepsilon^{(\nu)})=\prod_{k=1,\neq j}^{m}\left(1+\frac{\varepsilon_k^{(\nu)}}{\zeta_j+\varepsilon_j^{(\nu)}-z_k^{(\nu)}}\right)+O(\varepsilon^{(\nu)})\\&=\prod_{k=1,\neq j}^{m}\left(1+\frac{\varepsilon_k^{(\nu)}}{\zeta_j-z_k^{(\nu)}}+O(\varepsilon_k^{(\nu)}\varepsilon_j^{(\nu)})\right)+O(\varepsilon^{(\nu)})=1+\sum_{k=1,\neq j}^{m}\frac{\varepsilon_k^{(\nu)}}{\zeta_j-z_k^{(\nu)}}+O(\varepsilon^{(\nu)}).\end{aligned} \tag{9}$$

よって, 式 (9) を式 (7) に代入することにより, $\varepsilon_j^{(\nu+1)}=O((\varepsilon^{(\nu)})^2)$ を得る. ∎

4 算法の実際と計算例

具体的に算法を構成するには、次の2点に注意する必要がある。

a) 近接根に対応する $\tilde{P}(x)$ の初期値の選択。

b) $\tilde{P}(z_j^{(\nu)})$ と $Q'_\nu(z_j^{(\nu)})$ の計算の効率化。

a) に関しては、$\tilde{P}(x)$ の m 次以下の項からなる式

$$\tilde{c}_m x^m + \cdots + \tilde{c}_0 \tag{10}$$

の根を（通常の）Durand-Kerner 法で計算し、それらを近接根用 Durand-Kerner 法の初期値に設定する。（仮定より、求める近接根以外の根は複素平面上の単位円から十分離れており、式 (10) の根は求める近接根に近い位置に存在する。）

b) に関しては、係数に関する性質 (4) に着目する。$\hat{P}_\nu(x) = \prod_{j=1}^{m}(x - z_j^{(\nu)})$ とおくと、$z_1^{(\nu)}, \cdots, z_m^{(\nu)}$ が半径2程度の複素円盤内にあることから、$\hat{P}_\nu(x)$ の各項の係数は1よりそれほど大きくはない。したがって、

$$Q_\nu(x) = \mathrm{quo}(\tilde{P}(x), \hat{P}_\nu(x)) \ = \ q_{n-m}^{(\nu)} x^{n-m} + \cdots + q_1^{(\nu)} x + q_0^{(\nu)} \tag{11}$$

とおくと、$|q_j^{(\nu)}| = o(\gamma^{j-1})$ $(j = 1, 2, \ldots, n-m)$ と見做せる。

反復算式 (6) では、$\tilde{P}(z_j^{(\nu)})$ と $Q'_\nu(z_j^{(\nu)})$ さえ精度よく計算できればよいが、$|z_j^{(\nu)}| \lesssim 2$ なので、$\tilde{P}(x)$ と $Q'_\nu(x)$ の高次項は計算結果にはほとんど影響しない。したがって、$\tilde{P}(x)$ と $Q_\nu(x)$ はそれぞれ $\mu + m$ 次、μ 次以下の項を計算しておけば十分である。ここで、μ は数値根に必要な精度と γ で決まる整数である。たとえば、倍精度計算で $\gamma = 0.1$ の場合、$\mu = 20$ 程度で十分である。

次に計算例を示す。（この計算は数式処理システム GAL を用いて行った。）

例 1

$\delta = 1.0 \times 10^{-4}$ に対し、

$$\begin{aligned} P(x) = {} & (x - 1.2 + \delta)(x - 1.2)(x - 1.2 - \delta)(x - 1.2 - 2\delta) \\ & \times (x + 1)\{(x + 2)^2 + 2\}\{(x - 5)^2 + 1\}\{(x + 5)^2 + 3\}\{(x - 7)^2 + 3\}(x - 10) \end{aligned} \tag{12}$$

とおき、$x = 1.2$ 付近の近接根を計算する。多倍長浮動小数演算の精度は10進34桁（IEEE 倍精度の約2倍）とする。多倍長演算で近似無平方分解を行うと、$x - 1.2000499725317502515749723530879445 0$ が $P(x)$ の近似4重因子として計算される。そこで、多倍長演算を用いて、$x = 1.2000499725317502515\cdots$ の位置に $P(x)$ の原点を移動し、4個の近接根が原点を中心に半径2の円盤内に含まれるよう、スケール変換と規格化を行う。このことにより、式 (2) に対応する多項式 $\tilde{P}(x)$ として次式を得る。

$$\begin{aligned} \tilde{P}(x) = {} & -1.7578537183681 \times 10^{-45} x^{14} + \cdots + 4.0748102580441 \times 10^{-22} x^9 \\ & + 1.0683056800151 \times 10^{-17} x^8 - 2.3187945753916 \times 10^{-13} x^7 - 5.0470051845535 \times 10^{-9} x^6 \\ & + 5.599830886856 \times 10^{-5} x^5 + x^4 - 0.00075089624770413 x^3 - 1.0 x^2 \\ & + 0.0003524888550432 x + 0.090000002786884. \end{aligned} \tag{13}$$

式 (13) の $\tilde{P}(x)$ の5次以上の係数は、次数が1増加するたびに約 10^{-4} 倍ずつ小さくなっていくのが見てとれる。$\tilde{P}(x)$ の9次以上の項は、係数の大きさは 10^{-20} より小さく、反復式 (6) における $\tilde{P}(z_j^{(\nu)})$ の計算には影響しない。

$\tilde{P}(x)$ の 4 次以下の項からなる式の根を通常の Durand-Kerner 法で求めると、反復計算は 8 回で収束し、根として

$$\begin{aligned} x &= 0.31640116584382 - 2.9387358770557 \times 10^{-39} i, \\ x &= -0.94848132719237, \\ x &= -0.31605441686571, \\ x &= 0.94888547446197 \end{aligned} \tag{14}$$

を得る（i は虚数単位を表す）。（これらの根は、与多項式において、それぞれ

$$\begin{aligned} x &= 1.2000999999445 - 4.6465490204057 \times 10^{-43} i, \\ x &= 1.1999000044786, \\ x &= 1.1999999999448, \\ x &= 1.2002000044862 \end{aligned} \tag{15}$$

を表す。）

次に、式 (14) を初期値として近接根用 Durand-Kerner 法を実行する。反復計算に用いる $\tilde{P}(x)$ としては、式 (13) の 8 次以下の項のみを用いる。同様に、反復式 (6) の初期値に対し、式 (11) において $\nu = 0$ とおいた多項式 $Q_0(x)$ として、次に掲げる 4 次以下の項のみからなる式を用いる。

$$\begin{aligned} Q_0(x) = {} & 1.0683056800151 \times 10^{-17} x^4 \\ & + (-2.318794495173 \times 10^{-13} - 3.1394682295228 \times 10^{-56} i) x^3 \\ & + (-0.000000050470053479879 + 6.8142252413433 \times 10^{-52} i) x^2 \\ & + (0.000055998304846903 + 1.4832031290705 \times 10^{-47} i) x \\ & + (1.0000000370019 - 1.645595346359 \times 10^{-43} i). \end{aligned} \tag{16}$$

この結果、近接根用 Durand-Kerner 法は反復回数 3 回で収束し、近接根として

$$\begin{aligned} x &= 1.2001000000000000 - 2.9323846931128 \times 10^{-72} i, \\ x &= 1.1999000000000000 - 7.3309617327821 \times 10^{-73} i, \\ x &= 1.2000000000000000, \\ x &= 1.2002000000000000 + 2.9323846931128 \times 10^{-72} i \end{aligned} \tag{17}$$

を得る。

5 実験

以上で述べた算法の効率を確かめるための実験を行った。実験には SPARCStation 5 (microSPARC II 85MHz, RAM 32MB, SunOS 4.1.4) 上で数式処理システム GAL を用いた。

本実験では、1 つの近接根クラスタをもつ多項式を係数を乱数で与え、それらの近接根を計算した。多項式の次数は 10 次から 50 次まで 10 次おきに定め、それぞれの次数において多項式を 10 個ずつ生成した。近接根の多重度は 4、近接度は 10^{-4} 程度とし、近接根クラスタの中心は複素平面上の単位円盤内（かつその虚部は実軸から近接度程度の範囲内）に乱数で与えた。

近接根用 Durand-Kerner 法の実行にあたっては、4 で述べた初期値の選択 (a) および算法の効率化 (b) を行った場合と行わなかった場合（以下ではそれぞれ「効率化あり」「効率化なし」と呼ぶ）の計算時間を

次数	効率化なし	効率化あり
10	38.0	26.8
20	73.6	36.4
30	136.0	62.6
40	222.6	102.8
50	282.8	152.0

表 1: 効率化なし/ありのそれぞれの場合の, 各次数における 1 多項式あたりの平均計算時間（単位は ms, 以下同様）.

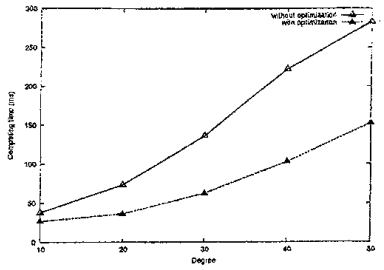

図 1: 表 1 のグラフ（実線/破線は, それぞれ効率化なし/ありの場合を表す. 以下同様）.

次数	効率化なし	効率化あり			
	反復回数	n_1	n_2	d_1	d_2
10	10.4	8.3	3.0	10.0	6.0
20	10.5	8.8	3.0	9.9	5.9
30	11.0	8.2	3.0	10.5	6.5
40	11.4	7.9	3.0	11.8	7.8
50	10.5	8.4	3.0	10.6	6.6

表 2: 効率化なしの場合における近接根用 D-K 法の平均反復回数と, 効率化ありの場合における初期値計算用/近接根用 D-K 法の平均反復回数（それぞれ n_1, n_2）, D-K 反復に用いた $\tilde{P}(x)$, $Q'_\nu(x)$ の低次項の平均次数（それぞれ d_1, d_2）.

比較した。計算時間は、近接根クラスタの中心を計算する近似無平方分解の時間を除き、原点移動と近接根用 Durand-Kerner 法の実行時間を測定した。ガーベジコレクション等の時間を除いた実計算時間を測定し、各次数毎に多項式毎の計算時間の平均値を算出、比較した。

5.1 実験 1

効率化なし/ありのそれぞれの場合の各次数における 1 多項式あたりの平均計算時間（単位：msec.）を表 1 および図 1 に示す。

次に、効率化なし/ありのそれぞれの場合における Durand-Kerner 法の平均反復回数を表 2 に示す。表中、効率化なしの場合における反復回数は、近接根用 Durand-Kerner 法の平均反復回数を表す。一方、効率化ありの場合においては、以下の記号を用いている。

n_1 初期値計算用 Durand-Kerner 法の平均反復回数

n_2 近接根用 Durand-Kerner 法の平均反復回数

d_1 初期値計算用/近接根用 Durand-Kerner 法において、反復公式の分子の計算に用いた $\tilde{P}(x)$ の低次項の平均次数

d_2 近接根用 Durand-Kerner 法において、反復公式の分母の計算に用いた $Q'_\nu(x)$ の低次項の平均次数

効率化なしの場合は、入力多項式の次数が変化しても近接根用 Durand-Kerner 法の反復回数がほぼ一定である。一方、効率化ありの場合は、入力多項式の次数の増大にかかわらず、初期値計算用/近接根用

次数	効率化なし	効率化あり
10	30.4	18.6
20	53.8	16.6
30	93.6	17.2
40	135.6	15.2
50	153.4	17.4

表 3: 効率化あり/なしのそれぞれの場合の, 各次数における D-K 法の平均計算時間.

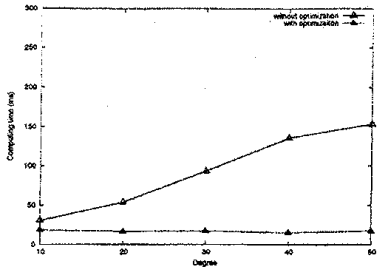

図 2: 表 3 のグラフ.

次数	効率化なし	効率化あり	平均値
10	2.8	3.8	3.3
20	13.2	14.0	13.6
30	33.2	34.8	34.0
40	71.8	69.8	70.8
50	112.8	116.8	114.8

表 4: 各次数における原点移動時間. 効率化あり/なしのそれぞれの場合および両者の平均値を表す.

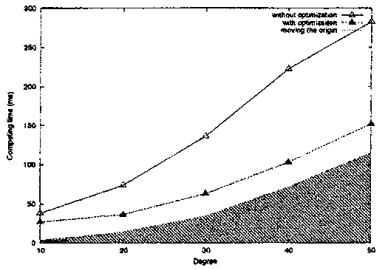

図 3: 図 1 に表 4 の平均値を重ねたグラフ. 灰色の部分は表 4 の平均値を示す.

Durand-Kerner 法の反復公式に用いられる多項式の次数が分母、分子ともほぼ一定で、かつ反復回数もほぼ一定であるので、Durand-Kerner 法の計算量は、入力多項式の次数にかかわらずほとんど変化しないはずである。それにもかかわらず、表 1、図 1 の通り、全体の計算時間は入力多項式の次数の増加に伴って増えている。

そこで、Durand-Kerner 法の実行時間のみを取り出したところ、表 3 および図 2 の通りになった。Durand-Kerner 法の計算時間は表 2 の反復回数をほぼ反映したものである。

さらに計算時間の内訳を調べたところ、近接根クラスタを原点付近に移動する原点移動の時間が、計算時間全体において大きな割合を占めていることがわかった。結果を表 4 および図 3 に示す（図 3 の灰色で塗りつぶされた部分が、効率化なし/ありのそれぞれの場合の原点移動時間の平均値を示す）。多項式の原点移動は GAL の多倍長浮動小数演算（精度 10 進 34 桁）で行っており、多項式の次数が増大するほど原点移動の時間が計算時間全体に影響することがわかる。

5.2 実験 2: QD Library を用いた原点移動の効率化

そこで、原点移動における多倍長浮動小数演算に外部のライブラリを用いることにより、原点移動の効率化を図った。

多倍長浮動小数演算には QD Library [3] と呼ばれるライブラリを用いた。QD Library は IEEE 倍精度の約 2 倍と約 4 倍の精度（以下、それぞれ 4 倍精度、8 倍精度と呼ぶ）の浮動小数演算を C++で実装したライブラリで、C++の他に Fortran-90 用のインターフェースが用意されている。ソースコードは公開されており、(著者らによる調査の限りライセンス条項は不明であるが) 無償で使用できる。

本実験では、要求される浮動小数演算の精度が IEEE 倍精度の約 2 倍の精度であるため、QD Library から 4 倍精度のライブラリを用いた。GAL は外部ライブラリの関数を呼び出すことができるので、QD Library

次数	効率化なし	効率化あり	原点移動
10	37.8	26.2	3.5
20	74.8	30.6	6.8
30	125.8	39.2	12.8
40	173.4	55.2	17.1
50	210.0	70.0	25.9

表 5: 効率化なし/ありのそれぞれの場合の, 各次数における 1 多項式あたりの平均計算時間と原点移動の平均時間（QD Library を使用）.

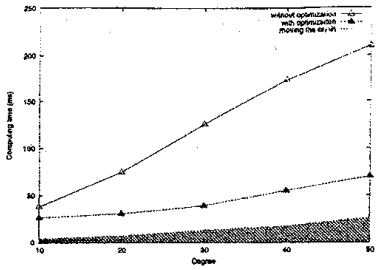

図 4: 表 5 のグラフ. 灰色の部分は原点移動の平均時間を表す.

を用いて原点移動のライブラリを作成し、GAL から同ライブラリを呼び出すことにより、原点移動を行った。そして、実験 1 と同様の実験を行い、計算時間を測定、比較した。

実験結果を表 5 および図 4 に示す。本実験結果の通り、原点移動を効率化することにより、4 で述べた効率化の本来の効果が現われたことがわかる。

参 考 文 献

[1] E. Durand. *Solutions Numériques des Équations Algébriques,* Tome I. Masson, Paris, 1960.

[2] S. Fortune. An iterated eigenvalue algorithm for approximating roots of univariate polynomials. *J. Symbolic Comput.*, Vol. 33, No. 5, pp. 627–646, 2002.

[3] Y. Hida, X. S. Li, and D. H. Bailey. Algorithms for quad-double precision floating point arithmetic. In *Proc. the 15th IEEE Symposium on Computer Arithmetic (ARITH'01)*, pp. 155–162, 2001. Software can be obtained from Hida's home page: `http://www.cs.berkeley.edu/~yozo/`

[4] I. O. Kerner. Ein Gesamtschrittverfahren zur Berechnung der Nullstellen von Polynomen. *Numer. Math.*, Vol. 8, pp. 290–294, 1966.

[5] X.-M. Niu and T. Sakurai. A method for finding the zeros of polynomials using a companion matrix. *Japan J. Indust. Appl. Math.*, Vol. 20, pp. 239–256, 2003.

[6] M.-T. Noda and T. Sasaki. Approximate GCD and its application to ill-conditioned algebraic equations. *J. Comput. Appl. Math.*, Vol. 38, No. 1-3, pp. 335–351, 1991.

[7] V. Y. Pan. Univariate polynomials: nearly optimal algorithms for numerical factorization and root-finding. *J. Symbolic Comput.*, Vol. 33, No. 5, pp. 701–733, 2002.

[8] T. Sasaki and M.-T. Noda. Approximate square-free decomposition and root-finding of ill-conditioned algebraic equations. *J. Inform. Process.*, Vol. 12, No. 2, pp. 159–168, 1989.

[9] A. Terui and T. Sasaki. "Approximate zero-points" of real univariate polynomial with large error terms. 情報処理学会論文誌, Vol. 41, No. 4, pp. 974–989, April 2000.

[10] A. Terui and T. Sasaki. Durand-Kerner method for the real roots. *Japan J. Indus. Appl. Math.*, Vol. 19, No. 1, pp. 19–38, January 2002.

[11] J.-C. Yakoubsohn. Finding a cluster of zeros of univariate polynomials. *J. Complexity*, Vol. 16, No. 3, pp. 603–638, 2000.